



InterActive 2

Written By: Steven Robert Cypherd

SUMMARY

This is an explanation of the code for the flow control I came up with it to get my Basic Stamp 2(BS2) talking to my Basic Stamp 1(BS1). I am showing the main loops only. For the code files look at letsmakerobots.com and instructables.com. Files or a zip file.

In my code I use a lot of Bit flags. Most processors allow access to the Bits of a Byte or Word to use like regular variables. Basic Stamp 1 and most Picaxe processors have defined variable names for the Bits of first few bytes of memory. Usually it is Bit0 – Bit7 or Bit0 – B15 or Bit0 – Bit31 to define a variable. In both of these processors the variable Bits can also be in another variable. Like my lights Byte B0 the Bits are the variables Bit0 – 7 that are lit1 – lit8 the individual lights. This makes testing of a light easy. “If lit1 = 1 then litOn” is simple.

The Basic Stamp 2 and processors like it use defined subscripting to access the Bits, Nibbles and Bytes of a variable. My individual lights are the variables "lights.Bit0 - Bit7". In testing a variable Bit is “if lgths.Bit0 = 1 then litOn”. Setting is “litTmp = lights.Bit0”. Usually the subscripts can not be a variable. Look at the program files in the EasyLights article. Read your manual.

I fixed as much as I could. The BS1 doesn't do serial out very well. It is stable and runs like any simple button on a system like this. The BS1 doesn't do Bits very well either.

My flow control is the only way I got these systems to communicate.

I uploaded full code pictures.

Interactive 2

An Explanation of my code for my Flow Control.

Slave R01

Master R02

P4 Blue < White Tx P4

P5 Tx > Black Rx P4

P2 Button 1

Vin

Vin

Green

P3 Button 2

G

G

Green

Serial to my Arduino Uno

Rx Yellow < Tx White

Tx Blue > Rx Black

G Black

G Green

Busy 1 Pinout

Work in Progress

[illegible]

```

end = 1
input = 0
whbase = 100
Abase = 2
NSR = 4

def:
  if modbase = 1 1/4th p2
    p = 1/4th = 1/4th p2
    GO TO p2

Basic Stamp 1 Panel board(BSP1)
Abase = 0
Use Message Acknowledge
PULSE 6
NSR pulse, 296, 350, eni, p1a1/2, CR
PULSE 3
NSR 4
en1/4CR
The BT1 (BJT) does not need any wire.
if a1/2 = 10 1/4th p2
  p = 1/4th = 1/4th p2
  en1/2 = 0
  GO TO p2

en1:
  DEBUT: Stop BT1 : DEC a1/2/4 : 1, 8, BN a1/2/4, CR
  DEBUT: Stop Input : DEC input/1 : 8, BN input, CR
  if a1/2 = 10 1/4th p2
    p = 1/4th = 1/4th p2
    en1/2 = 0
    GO TO p2

en2:
  Note in progress
  DEBUT: Wn2 : DEC a1/2/4 : 1, 8, BN a1/2/4, CR
  en2/2 = 1
  GO TO p2

```

- Last page of BS2 I2
- BS1 I2 code.

[illegible][illegible]

```

nrd
  f obj = 1 then p1 Check for message
  send obj, 12450, mCmd, tmp1
  flush 3
  f obj = 1 then p1 Check for correct address
  obj = tmp1
  do my own assignments
  for (int i = 0; i < 7; i++)
    tmp1 = obj1
    BunchObj tmp1, obj1, obj1, obj1, obj1, obj1, obj1, obj1
    obj1 tmp1 = 80 GOCD doneCheck
    obj1 tmp1 = 85 GOCD doneCheck
    obj1 tmp1 = 86 GOCD doneCheck
    obj1 tmp1 = 88 GOCD doneCheck
    obj1 tmp1 = 89 GOCD doneCheck
    obj1 tmp1 = 87 GOCD doneCheck
    obj1 tmp1 = 88
    doneCheck
  if obj = 1 then doObjOn: a simple test
  objOn = obj1, obj1, obj1, obj1
  objOn =
  objOn
  pClock = objOn
  objOn = obj1
  next
end
f obj = 1 then p3
f obj = 1 then p3
obj1 obj = 0
end
GOCD run
end

```

- Last page of BS2 I2
- BS1 I2 code.

Step 4

[illegible]

- First working version BS2 I1
- First working version BS1 I1

Step 4

```

run2:Look for message from the master
IF sSet = 0 THEN jp1 Set Up
IF pRx = 1 THEN jp1 Check for message
SERIN pinRx, T2400, mCmd, tmp1
PAUSE 20
IF mCmd <> BS1P1 THEN jp1 Check for correct address
lights = tmp1
GOSUB dolights
jp1:Clear buttons
IF pBtn1 = 1 THEN jp2
IF pBtn2 = 1 THEN jp2
bDown = 0
btn1 = 0
btn2 = 0
jp2:Set up slave
IF sSet = 1 THEN jp3
tmp1 = tmp1 + 1
IF tmp1 < 10 THEN jp3
sSet = 1
jp3:
GOTO run1t
END

'Subroutines *****
dolights: From EasyLights article
FOR tmp1 = 0 TO 7
  litmp = tmp1
  GOSUB checkLights
  IF litmp = 1 THEN doLitOn 'A simple test
  pData = litOff : GOTO doLitCk
  doLitOn:
  pData = litOn
  doLitCk:
  pClock = litOn
  pClock = litOff
  NEXT
RETURN

Each Lit is 1 for on and 0 for off
checkLights: 'An Array for anything
BRANCH litmp, ckLit1, ckLit2, ckLit3, ckLit4, ckLit5, ckLit6, ckLit7, ckLit8)
ckLit1: litmp = lit1 : GOTO doneCheckIt
ckLit2: litmp = lit2 : GOTO doneCheckIt
ckLit3: litmp = lit3 : GOTO doneCheckIt
ckLit4: litmp = lit4 : GOTO doneCheckIt
ckLit5: litmp = lit5 : GOTO doneCheckIt
ckLit6: litmp = lit6 : GOTO doneCheckIt
ckLit7: litmp = lit7 : GOTO doneCheckIt
ckLit8: litmp = lit8
doneCheckIt:
RETURN

```

- Last page BS1 I1

Get them Talking.

This document was last generated on 2012-11-02 04:41:11 AM.